CALPA-NET: Channel-pruning-assisted Deep Residual Network for Steganalysis of Digital Images

Shunquan Tan, *Senior Member, IEEE,* Weilong Wu, Zilong Shao, Qiushi Li, Bin Li, *Senior Member, IEEE,* and Jiwu Huang, *Fellow, IEEE*

***Abstract*—Over the past few years, detection performance improvements of deep-learning based steganalyzers have been usually achieved through structure expansion. However, excessive expanded structure results in huge computational cost, storage overheads, and consequently difficulty in training and deployment. In this paper we propose CALPA-NET, a ChAnneL-Pruning-Assisted deep residual network architecture search approach to shrink the network structure of existing vast, over-parameterized deep-learning based steganalyzers. We observe that the broad inverted-pyramid structure of existing deep-learning based steganalyzers might contradict the well-established model diversity oriented philosophy, and therefore is not suitable for steganalysis. Then a hybrid criterion combined with two network pruning schemes is introduced to adaptively shrink every involved convolutional layer in a data-driven manner. The resulting network architecture presents a slender bottleneck-like structure. We have conducted extensive experiments on BOSSBase+BOWS2 dataset, more diverse ALASKA dataset and even a large-scale subset extracted from ImageNet CLS-LOC dataset. The experimental results show that the model structure generated by our proposed CALPA-NET can achieve comparative performance with less than two percent of parameters and about one third FLOPs compared to the original steganalytic model. The new model possesses even better adaptivity, transferability, and scalability.**

***Index Terms*—steganalysis, steganography, deep learning, convolutional neural network, network pruning.**

I. Introduction

STEGANALYSIS aims to detect covert communication established via steganography. In addition to detection performance, computational cost, model complexity, as well as adaptivity, transferability, and scalability, are all important considerations for a real-world steganalytic framework.

Over the last decade, the main battleground of the war between modern steganography and steganalysis has always been in digital images [1]. Most of the spatial-domain and frequency-domain image steganographic algorithms have adopted the embedding distortion minimizing framework [2]. The prominent additive embedding distortion minimizing schemes include HILL [3] and MiPOD [4] in spatial domain, as well as UERD [5] in JPEG domain. UNIWARD [6] is a benchmarking cross-domain scheme (noted as S-UNIWARD in spatial domain, while J-UNIWARD in JPEG domain).

In recent years, deep-learning frameworks, especially Convolutional Neural Networks (CNNs) have achieved overwhelming superiority over conventional approaches in many fields [7]. In the meanwhile, from early AlexNet [8] and VGGNet [9], to later more advanced Inception models [10] and ResNet [11], the literature witnessed deep-learning frameworks have become more and more deeper and complicated.

The once overlord of image steganalysis is the “rich model” hand-crafted features family [12]–[16] equipped with an ensemble classifier [17]. Started from the work of Tan and Li [18], breakthroughs were made in deep-learning based steganalysis [19]–[21]. Then Ye et al. proposed a deep-learning based steganalyzer equipped with a new activation function called Truncated Linear Unit (TLU) [22], achieving significant improvement compared to those established “rich model” steganalytic features in spatial domain. In JPEG domain, deep-learning based steganalyzers have also outperformed the “rich model” features family. Chen et al. proposed a specific deep-learning based steganalyzer aware of JPEG phase [23]. Xu proposed a novel 20-layer framework with residual connections (named XuNet2) [24]. Aiming at large-scale JPEG image steganalysis, Zeng et al. proposed a generic hybrid deep-learning framework incorporating the domain knowledge behind rich steganalytic models [25]. On the contrary, Boroumand et al. proposed a deep residual steganalytic network designed to minimize the use of heuristic domain knowledge (named SRNet) [26], which has shown superior detection performance for both spatial domain and JPEG domain steganography. In [27], Zeng et al. explicitly considered the correlation among color bands and proposed WISERNet, the wider separate-then-reunion network specifically designed for steganalysis of true-color images. For a comprehensive survey please refer to [28].

It has been widely acknowledged that existing deep-learning frameworks are over parameterized, and are therefore with huge computational cost and storage overheads. Consequently, they are more likely to require massive amounts of training data to achieve good performance, and are hard to deploy [29]. As also discussed in [28], automatic network architecture searching techniques [30] might be potential to build up an effective deep-learning framework. Another trend is network pruning, which is one of the most popular methods to reduce network complexity. Network pruning has become one

S. Tan, W. Wu, Z. Shao, and Q. Li are with College of Computer Science and Software Engineering, Shenzhen University. B. Li and J. Huang are with College of Electronic and Information Engineering, Shenzhen University.

All of the members are with the Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen Key Laboratory of Media Security, Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen Institute of Artificial Intelligence and Robotics for Society, China (email: tansq@szu.edu.cn).

This work was supported in part by the Key-Area Research and Development Program of Guangdong Province (2019B010139003), NSFC (61772349, U19B2022, U1636202, 61872244), Guangdong Basic and Applied Basic Research Foundation (2019B151502001), and Shenzhen R&D Program (GJHZ20180928155814437, JCYJ20180305124325555).

important research problem even since a very early stage of the evolution of deep-learning frameworks [31]. For CNNs, state-of-the-art network pruning approaches can be classified into two categories, the non-structured weights pruning and the channel-based structured pruning methods. Non-structured weights pruning methods [32] cannot lead to complexity reduction without the support of dedicated hardware. Therefore structured pruning methods are more practical. Hu et al. proposed a channel pruning method based on the percentage of zeros in the outputs [33]. Li et al. proposed another channel pruning method based on the l_1 norms of the corresponding filter weights [34]. Luo et al. proposed ThiNet which greedily prunes those channels with smallest effects on the activation values of the next layer [35]. Further on, Huang and Wang used sparsity regularization to prune channels and even coarser structures, such as residual blocks [36]. The works mentioned above are just the notable ones selected from a complete collection. But, whatever the approach, the existing network pruning approaches have all followed a typical three-stage pipeline: training, pruning, and then finetuning in order to preserve a set of inherited learned important weights.

In this paper, we propose CALPA-NET, a channel-pruning-assisted deep residual network architecture search approach to shrink the network structure of existing vast, over-parameterized deep-learning based steganalyzers. We observe that the established “doubling the number of channels along with halving the size of output feature maps” rule might undermine the diversity of output features of deep steganalytic models. Therefore starting from existing bloated deep steganalytic models, CALPA-NET utilizes a hybrid criterion to adaptively determine the number of channels of every involved convolutional layer. The proposed hybrid criterion combines the ThiNet scheme [35] and the l_1-norm based scheme [34]. Please note that CALPA-NET, where we abandon the training-pruning-finetuning pipeline of typical network pruning methods, is entirely different from network pruning approaches mentioned above since the proposed channel pruning criterion is just used to search efficient network architectures. The extensive experiments conducted on public datasets show that even trained from scratch, the shrunken model can still achieve state-of-the-art detection performance with merely a tiny proportion of parameters and much lower computational complexity.

The rest of the paper is organized as follows. Sect. II firstly gives a brief overview of existing representative deep steganalytic models as well as channel-based structured pruning methods. Then experimental testimonies are provided to support the rationale of CALPA-NET. Next the procedure with our proposed hybrid channel pruning criterion used in CALPA-NET is described in detail. Results of experiments conducted on public datasets are presented in Sect. III. Finally, we make a conclusion in Sect. IV.

II. Our proposed CALPA-NET

A. Preliminaries

As far as we know, existing deep-learning based steganalyzers are all based on CNNs (Convolutional Neural Network). The principal part of CNN can be modeled as a direct graph of alternating convolutional layers and auxiliary layers (e.g. BN (Batch Normalization) layers and pooling layers). Convolutional layers are the central components of a CNN since they contain the overwhelming majority of learnable weights and biases. For a given convolutional layer L_l, it takes an after-activation input tensor $\widehat{\mathcal{Z}}^{l-1} \in \mathbb{R}^{J^{l-1} \times H^{l-1} \times W^{l-1}}$ which possesses J^{l-1} input channels with height H^{l-1} and width W^{l-1}, convolves it with $\mathcal{W}^l \in \mathbb{R}^{J^{l-1} \times K^l \times W^l \times W^l}$, a filter tensor consisting of $J^{l-1} \times K^l$ kernels with size $W^l \times W^l$, and generates $\mathcal{Z}^l \in \mathbb{R}^{K^l \times H^l \times W^l}$, the corresponding before-activation output tensor which has K^l output channels (feature maps) with height H^l and width W^l. The convolution operation can be modeled as (the bias is omitted for brevity):

$$\mathcal{Z}^l_{k::} = \sum_{j=1}^{J^{l-1}} \widehat{\mathcal{Z}}^{l-1}_{j::} * \mathcal{W}^l_{jk::}, \ 1 \le k \le K^l \tag{1}$$

where $\widehat{\mathcal{Z}}^{l-1}_{j::}$, $\mathcal{W}^l_{jk::}$, and $\mathcal{Z}^l_{k::}$ denotes the *slides*, the two dimensional sections defined by fixing all but the last two indices, of $\widehat{\mathcal{Z}}^{l-1}$, $\mathcal{W}^l$, and $\mathcal{Z}^l$, respectively.

1) Interior structures of deep-learning based steganalyzers: Two popular deep-learning based steganalyzers, SRNet [26] and XuNet2 [24] are taken as examples. Their conceptual structures are illustrated in Fig. 1. The two deep-learning based steganalyzers both take spatial representation of the target image as input. All deep-learning based steganalyzers can be divided into three modules: the bottom module aiming at extracting the so-called “stego noise residuals”, the middle module which striving for learning compact representative features and the top module which is a simple binary classifier.

Actually most of the existing research works regarding to deep-learning based steganalyzers have been devoted to the bottom module. The bottom module of XuNet2 adopts a particular truncated filter bank with sixteen 4×4 DCT high-pass filters. Further on, the bottom module of SRNet consists of a pile of two hierarchical convolutional layers (“L1” and “L2”, following the notations in [26]) and five unpooled residuals blocks with direct shortcut connections (from “L3” to “L7”) for extraction of noise residuals. Conversely, according to our best knowledge, no specific domain knowledge has ever been introduced to guide the design of the middle module as well as the top module. Researchers just simply followed the effective recipes in other research fields (e.g. computer vision). Nowadays, almost all of the state-of-the-art deep-learning based steganalyzers, including SRNet and XuNet2, have adopted shortcut connections inspired by ResNet [11] and a simple design rule— “doubling the number of channels along with halving the spatial size of feature maps” which can be traced back to VGGNet [9]. Take SRNet for instance. As shown in Fig. 1, from “L3” up to “L11” are with shortcut connections. From its “L9” up to “L12”, the doubling numbers of output channels (64, 128, 256, and 512) correspond respectively to the halving sizes of feature maps (128×128, 64×64, 32×32, and 16×16).

2) Channel pruning methods: ThiNet and l_1-norm based: Since the training-pruning-finetuning pipeline of typical network pruning methods is completed abandoned in CALPA-

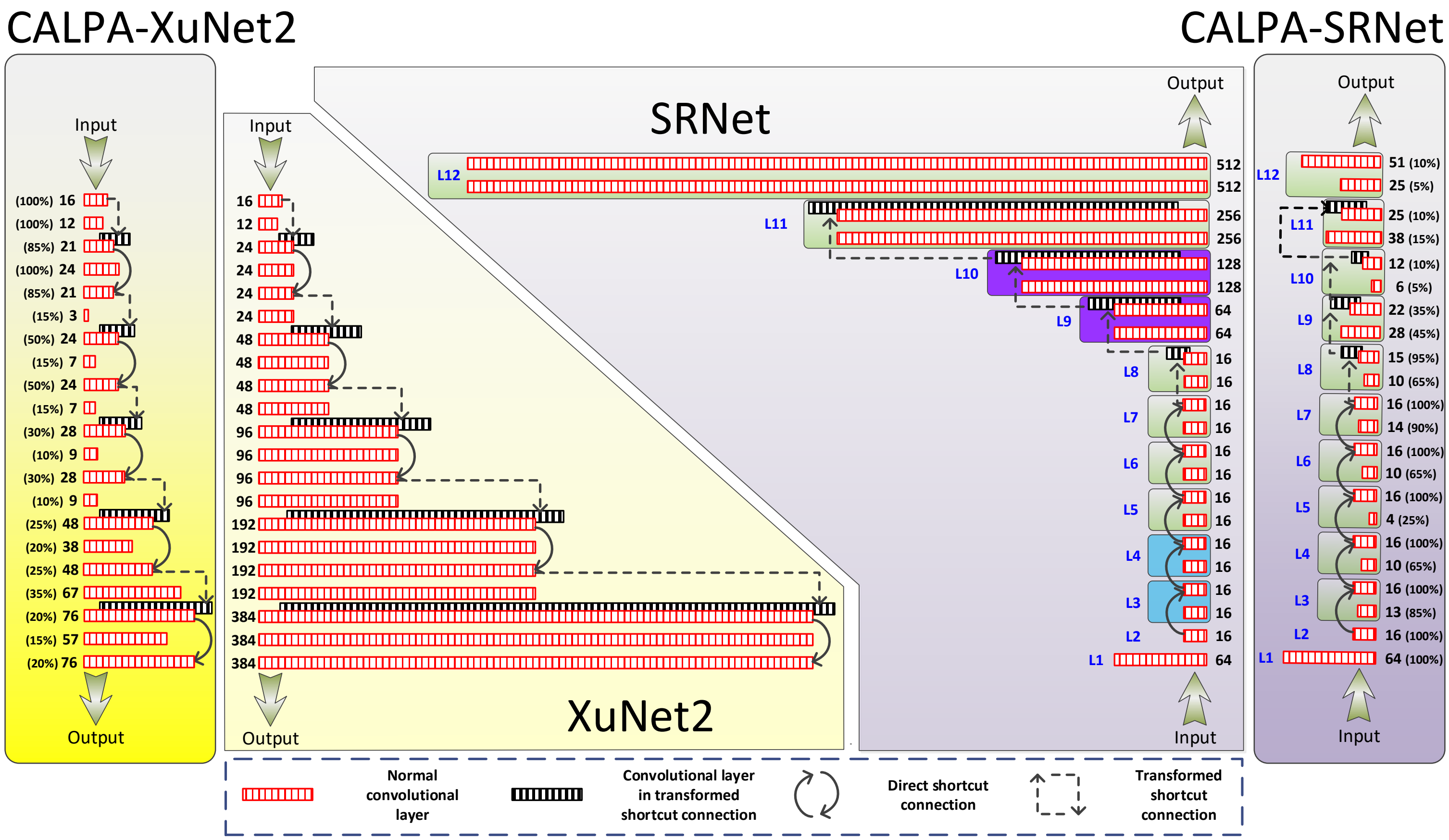


Fig. 1. From left to right, the conceptual structures of CALPA-XuNet2, XuNet2, SRNet, and CALPA-SRNet* are illustrated respectively. The number and corresponding shrinking rate of output channels of every convolutional layer is shown alongside the representing bar. For SRNet and CALPA-SRNet, blue "L1" to "L12" represent twelve composition blocks following the notations in [26].

* The shrinking rates of CALPA-XuNet2 and CALPA-SRNet are aiming at detecting J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75.

NET, only the pruning criteria in ThiNet [35] and the l_1-norm based scheme [34] are addressed.

ThiNet [35] is a data-driven channel selection algorithm. For a pre-defined pruning rate γ, a subset of m images are randomly selected from training set and fed to a trained CNN model one by one. Fed with every image in the subset, the input and output tensors of all the convolutional layers of the CNN model are obtained in a forward propagation. In ThiNet, in order to prune a given convolutional layer L_l, we have to observe its impact on the next layer L_{l+1}. Given a specific convolutional layer L_{l+1}, n elements are further randomly sampled from $\mathfrak{Z}^{l+1}$. Therefore a set with $\widetilde{m} = m \times n$ samples (namely, there are m images, and each image provides n elements for L_{l+1}) is obtained. We introduce a new index t to traverse those $\widetilde{m}$ elements, and denote the corresponding subset of *filters* in $\mathcal{W}^{l+1}$ as $\mathcal{W}^{l+1,t} \in \mathbb{R}^{J^l \times W^{l+1} \times W^{l+1}}$, $1 \leq t \leq \widetilde{m}$, and *receptive fields* [1] in $\widehat{\mathfrak{Z}}^l$ as $\widehat{\mathfrak{R}}^{l,t} \in \mathbb{R}^{J^l \times W^{l+1} \times W^{l+1}}$, $1 \leq t \leq \widetilde{m}$. What proposed in ThiNet is actual a greedy solution of the following optimization problem:

$$\underset{T}{\text{argmin}} \sum_{t=1}^{\widetilde{m}} \left(\sum_{j \in T} \widehat{\mathfrak{R}}^{l,t}_{j::} * \mathcal{W}^{l+1,t}_{j::} \right)^2$$

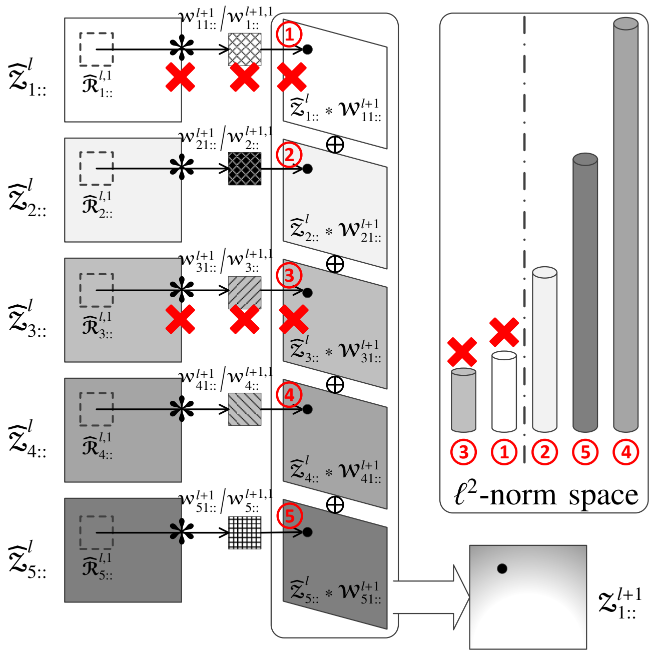

$\mathfrak{Z}^l_{1::}$
$\widehat{\mathfrak{R}}^{l,1}_{1::}$
$\mathcal{W}^{l+1}_{11::}$ / $\mathcal{W}^{l+1,1}_{1::}$
1
$\widehat{\mathfrak{Z}}^l_{1::} * \mathcal{W}^{l+1}_{11::}$
$\mathfrak{Z}^l_{2::}$
$\widehat{\mathfrak{R}}^{l,1}_{2::}$
$\mathcal{W}^{l+1}_{21::}$ / $\mathcal{W}^{l+1,1}_{2::}$
2
$\widehat{\mathfrak{Z}}^l_{2::} * \mathcal{W}^{l+1}_{21::}$
$\mathfrak{Z}^l_{3::}$
$\widehat{\mathfrak{R}}^{l,1}_{3::}$
$\mathcal{W}^{l+1}_{31::}$ / $\mathcal{W}^{l+1,1}_{3::}$
3
$\widehat{\mathfrak{Z}}^l_{3::} * \mathcal{W}^{l+1}_{31::}$
$\mathfrak{Z}^l_{4::}$
$\widehat{\mathfrak{R}}^{l,1}_{4::}$
$\mathcal{W}^{l+1}_{41::}$ / $\mathcal{W}^{l+1,1}_{4::}$
4
$\widehat{\mathfrak{Z}}^l_{4::} * \mathcal{W}^{l+1}_{41::}$
$\mathfrak{Z}^l_{5::}$
$\widehat{\mathfrak{R}}^{l,1}_{5::}$
$\mathcal{W}^{l+1}_{51::}$ / $\mathcal{W}^{l+1,1}_{5::}$
5
$\widehat{\mathfrak{Z}}^l_{5::} * \mathcal{W}^{l+1}_{51::}$
3 1 2 5 4
ℓ^2-norm space
$\mathfrak{Z}^{l+1}_{1::}$

Fig. 2. Illustration of ThiNet algorithm (Eq. (2)) via a toy example.

[1] The region of the input space that is relative to the unit we are taking into consideration.

$$\text{s.t.} \quad |T| = J^l \cdot \gamma,\ T \subset \{1, 2, \cdots, J^l\} \quad (2)$$

in which $|T|$ denotes the number of selected indexes in T, and $\widehat{\mathfrak{R}}^{l,t}_{j::}$, $\mathcal{W}^{l+1,t}_{j::}$ denote the j-th slide of $\widehat{\mathfrak{R}}^{l,t}$ and $\mathcal{W}^{l+1,t}$. Those $\widehat{\mathcal{Z}}^{l}_{j::}$ with indexes fall in T are categorized as "weak channels" and are pruned. The corresponding before-activation slides $\{\mathcal{Z}^{l}_{j::}, j \in T\}$ are implicated, and are pruned as well. Consequently, the corresponding filters $\{\mathcal{W}^{l}_{:j::}, j \in T\}$ which generate $\{\mathcal{Z}^{l}_{j::}, j \in T\}$ are also pruned. Please note that as three dimensional sub-tensors, the pruning of $\{\mathcal{W}^{l}_{:j::}, j \in T\}$ results in cutting down of numerous parameters and FLOPs (FLoating-point OPerations) in the corresponding convolutions.

Fig. 2 provides a toy example. In this toy example, the convolutional layer L_l to be pruned contains a before-activation output tensor $\mathcal{Z}^l$ with slide number $J^l = 5$. Therefore $\widehat{\mathcal{Z}}^{l}$, the corresponding after-activation input tensor for the next layer L_{l+1} contains five slides: $\widehat{\mathcal{Z}}^{l}_{1::}$, $\widehat{\mathcal{Z}}^{l}_{2::}$, $\widehat{\mathcal{Z}}^{l}_{3::}$, $\widehat{\mathcal{Z}}^{l}_{4::}$, and $\widehat{\mathcal{Z}}^{l}_{5::}$. The next layer L_{l+1} only contains $K^{l+1} = 1$ output channel/slide, corresponding to $\mathcal{Z}^{l+1}_{1::}$. The pre-defined pruning rate γ is set to 40%, which means that two out of the five slides in $\widehat{\mathcal{Z}}^{l}$ should be pruned. Only one point/element is randomly sampled from $\mathcal{Z}^{l+1}_{1::}$, the only one slide of L_{l+1} in this toy example. Therefore Eq.(2) of the manuscript can be simplified to:

$$\underset{T}{\operatorname{argmin}} \left(\sum_{j \in T} \widehat{\mathfrak{R}}^{l,1}_{j::} * \mathcal{W}^{l+1,1}_{j::} \right)^2, \text{s.t.} \quad |T| = 2,\ T \subset \{1, 2, \cdots, 5\}$$

From the equation above we can see $\mathcal{W}^{l+1,1}_{j::} = \mathcal{W}^{l+1}_{j1::}$, $1 \le j \le 5$, as also denoted in Fig. 2. For every filter, $\mathcal{W}^{l+1}_{j1::}$, $1 \le j \le 5$, the corresponding receptive field in $\widehat{\mathcal{Z}}^{l}$ is $\widehat{\mathfrak{R}}^{l,1}_{j::}$, $1 \le j \le 5$, respectively, as marked by dashed boxes in Fig. 2. Since the ℓ^2-norm of $\widehat{\mathfrak{R}}^{l,1}_{1::} * \mathcal{W}^{l+1,1}_{1::} + \widehat{\mathfrak{R}}^{l,1}_{3::} * \mathcal{W}^{l+1,1}_{3::}$ is minimal, they are decimated. As a result, $\mathcal{W}^{l+1,1}_{1::}$ and $\mathcal{W}^{l+1,1}_{3::}$ are pruned. Two slides, $\widehat{\mathcal{Z}}^{l}_{1::}$ and $\widehat{\mathcal{Z}}^{l}_{3::}$ which contain $\widehat{\mathfrak{R}}^{l,1}_{1::}$ and $\widehat{\mathfrak{R}}^{l,1}_{3::}$ are pruned as well. Though not illustrated in Fig. 2, we should note that in L_l, $\mathcal{Z}^{l}_{1::}$, $\mathcal{Z}^{l}_{3::}$, $\mathcal{W}^{l}_{:1::}$, $\mathcal{W}^{l}_{:3::}$ are also pruned accordingly.

On the contrary, l_1-norm based scheme [34] only considers the statistical properties of the filters themselves. For a given convolutional layer L_l, its K^l output channels are traversed. The l_1 norms of the corresponding filters $\{\mathcal{W}^{l}_{:k::}, 1 \le k \le K^l\}$ are calculated and sorted from lowest to highest. For a pre-defined pruning rate γ, the first $K^l \cdot \gamma$ filters in the sorted list are pruned. Those output channels corresponding to the pruned filters are removed as well.

B. Rationale of our proposed CALPA-NET

How to determine the width (number of output channels/trainable parameters) in each layer is a well-known open problem in deep learning literature. As far as we know, in the field of deep-learning based pattern recognition, there are no firm principles for determination of layer width given a fixed amount of computational resources. Therefore we opt to narrow our focus to deep-learning based image steganalysis.

In the field of image steganalysis, a notable philosophy is established since the reign of "rich model+ensemble classifier" solution [12]: model diversity is crucial to success of steganalytic detectors. The adoption of ResNet-style shortcut connections in latest deep-learning based steganalyzers actually supports the above philosophy since ResNet-style shortcuts resemble ensemble classifier, as pointed out in [37].

In fact, in the field of deep-learning based pattern recognition, upper-level representations (usually output channels of upper-level convolutional layers) learned with pre-trained deep CNNs are widely used as classification feature set [38], [39]. Since deep CNN based steganalysis is one of the subfields of deep-learning based pattern recognition, it would be reasonable to believe that the argument which makes sense in steganalytic feature-set construction can also be established for the mid-level representations of deep-learning based steganalyzers. Therefore, given a deep-learning based steganalyzer, its upper-level convolutional layer L_l can be regarded as a features extractor. Its every output channel (feature map) can be viewed as a generated sub-model and should contains diverse statistical patterns, according to [12].

From the perspective of time-frequency analysis, those ideal diverse statistical patterns correspond to sparse intensity fluctuations in nearly non-overlapping frequency subbands. Conversely, those strong intensities in the most common frequency subbands (usually those low-frequency subbands) usually represent the underlying universal patterns and contribute little to model diversity. However, as shown below, the experimental evidence clearly show that the aggregation in convolution with large enough input channels suppresses sparse intensity fluctuations as well as heightens strong intensities in low-frequency subbands.

We take a well-trained SRNet model aiming at detecting J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75 for sample. With the 256 × 256 JPEG representation of BOSSBase image No. 1013 as input, we investigate the output channels of the second convolutional layer of "Type 3" residual blocks ("L8", "L9", "L10", and "L11"). [2] For the broader "L11", we further investigate the intermediate feature maps before the summation of the representative output channels.

Fig. 3 shows the heat maps of amplitudes of some representative two-dimensional Fourier-transformed frequency-domain output channels for those residual blocks (please enlarge it for better visual effect). From Fig. 3 we can observe that there are strong intensities in the most common-frequency/low-frequency sub-bands of the outputs of the main branch of the residual blocks. Even for those output channels with drastic fluctuation, the evidence is obvious.

For "L11", three output channels of the second convolutional layer are further selected as samples. The left one in Fig. 3 is with relatively obvious intensity fluctuations in high-frequency subbands, the middle one only contains energy in low-frequency sub-bands, while the right one is in somewhere

[2] Please note that the output channels of the second convolutional layer of a given "Type 3" residual block, rather than the outputs of the entire residual block are investigated. This is due to the fact that with shortcut connection, the outputs of the entire "Type 3" residual block are the summation of the output channels of the second convolutional layer and the corresponding input channels of the entire residual block.

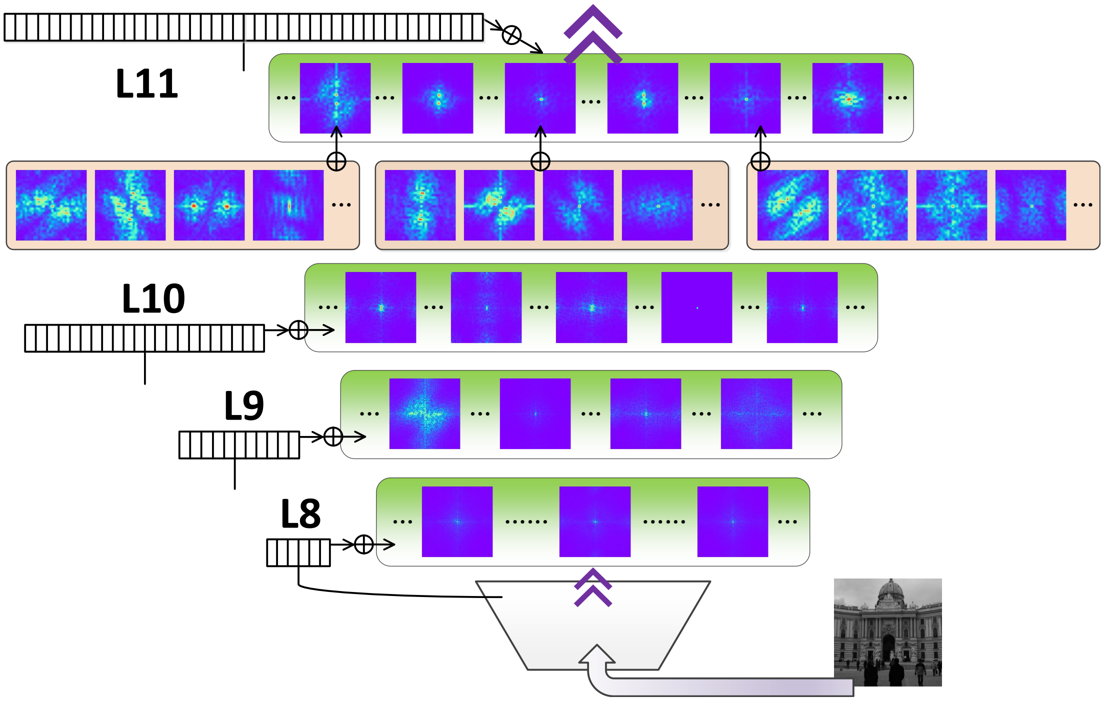


Fig. 3. Amplitudes of the two-dimensional Fourier-transformed output channels (visually represented with heat maps). The investigated target is a well-trained SRNet model aiming at detecting J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75. The input is BOSSBase image No. 1013. Output channels of the second convolutional layer of "Type 3" residual blocks ("L8", "L9", "L10", and "L11") are investigated. For "L11", we further investigate the intermediate feature maps before the summation of the representative output channels.

between the two extremes. For every one of the three output channels, four intermediate feature maps before the summation are further selected. All of those selected intermediate feature maps are with obvious intensity fluctuations in high-frequency subbands. From Fig. 3 we can see that with large enough input channels, for a given output channel, no matter how dispersive the resulting intensity fluctuations in it, the summation/aggregation in convolution of a given filter tensor do suppress the ever-present sparse intensity fluctuations in higher-frequency subbands of the intermediate feature maps generated by involved kernels.

Therefore we can make a straightforward inference that the broad convolutional layers in the inverted-pyramid style upper modules of existing deep-learning based steganalyzers might violate the model diversity oriented philosophy. In Appendix. A, we provide a theoretical reflection to support our argument.

C. Detailed algorithm of our proposed CALPA-NET

1) The overall procedure: Our proposed approach is named CALPA-NET, the ChAnneL-Pruning-Assisted deep residual NETwork for image steganalysis. "CALPA" is homophonic with "KALPA", a Sanskrit word for a never ending loop in which the universe circularly expands and shrinks. The demonstration in Sect. II-B has revealed that the broad inverted-pyramid structure might be a bloated solution for deep-learning based steganalyzers. Therefore our target is to shrink the excessive expanded structure of existing successful deep-learning based steganalyzers to make them more cost-effective, as the word "KALPA" implies.

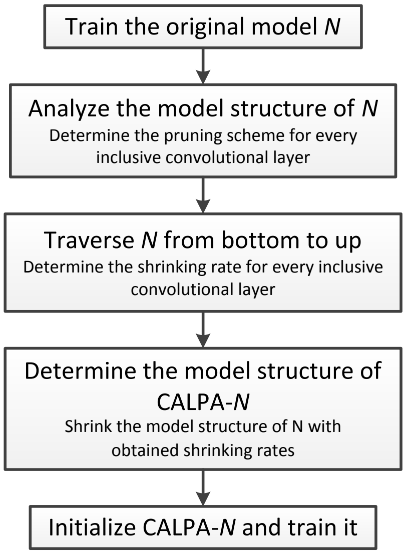

Train the original model *N*
Analyze the model structure of *N*
Determine the pruning scheme for every inclusive convolutional layer
Traverse *N* from bottom to up
Determine the shrinking rate for every inclusive convolutional layer
Determine the model structure of CALPA-*N*
Shrink the model structure of N with obtained shrinking rates
Initialize CALPA-*N* and train it

Fig. 4. The overall CALPA-NET diagram for a given deep-learning based steganalytic model *N*.

Recently, an interesting phenomenon that network pruning can be regarded as one sort of network architecture searching approach is reported in the field of computer vision [40]. An effective pruned network trained from scratch can show the performance as good as the one with traditional three-stage training-pruning-finetuning pipeline. This phenomenon inspires us to use effective network pruning criteria to determine the most cost-effective architecture for deep-learning based steganalyzers. The diagram of the overall procedure is

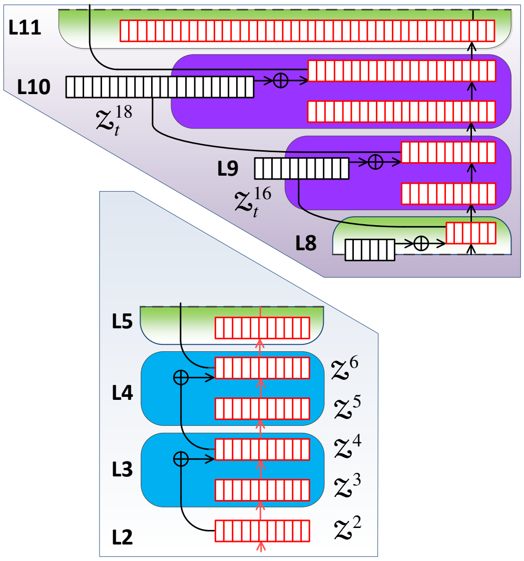


Fig. 5. The shortcut connections in two types of residual blocks in SRNet. "⊕" denotes element-wise addition.

shown in Fig. 4. Given a deep-learning based steganalytic model N, it is trained first using the original training protocol. At the same time, the structure of N is analyzed. A specific pruning scheme is determined for every inclusive convolutional layer in the pipeline according to its position in the pipeline (see Sect. II-C2). Then the well-trained N is traversed from bottom to top and the shrinking rate for every inclusive convolutional layer is determined in a data-driven manner. A brand new model structure is obtained via shrinking the channels of every inclusive layer with corresponding shrinking rate (see Sect. II-C3). The resulting model is referred to as CALPA-N, and is reset and trained from scratch. In this paper, two representative deep-learning based steganalytic models are involved: SRNet [26] and XuNet2 [24]. We call the corresponding CALPA-NET as CALPA-SRNet and CALPA-XuNet2.

2) The proposed hybrid channel pruning criterion: Unfortunately, there is no known channel pruning schemes for shortcut connections yet. Our preferred channel pruning criterion is the one used in ThiNet [35] due to its simplicity and efficiency. However, the ThiNet scheme cannot be used in shortcut connections.

An example from SRNet is given in Fig. 5. Two types of residual blocks are picked out from Fig. 1. Blocks "L3" and "L4" with blue background contain direct shortcut connections, while blocks "L9" and "L10" with pink background contain transformed shortcut connections. Please note that for direct shortcut connections, the element-wise addition of the two relevant convolutional layers demands that the sizes of the corresponding output tensors, like $\mathfrak{Z}^2$ and $\mathfrak{Z}^4$, $\mathfrak{Z}^4$ and $\mathfrak{Z}^6$, should be the same. Further more, all of the relevant convolutional layers linked via shortcut connections should be with the same output tensor size. From Fig. 5 it is clear that the size of $\mathfrak{Z}^2$, $\mathfrak{Z}^4$ and $\mathfrak{Z}^6$ must be the same. For transformed shortcut connections, each shortcut is equipped with a specific convolutional layer consisting of 1×1 kernels to extend the size of the shortcut output tensor. Given a main-branch output tensor $\mathfrak{Z}^l$, denote the output tensor in the transformed shortcut connection element-wisely added to it as $\mathfrak{Z}_t^l$. Analogously, the two convolutional layers relevant to the element-wise addition in transformed shortcut connections, like $\mathfrak{Z}^{16}$ and $\mathfrak{Z}_t^{16}$, $\mathfrak{Z}^{18}$ and $\mathfrak{Z}_t^{18}$, should also be the same.

In ThiNet, the pruning in $\mathfrak{Z}^l$ relies on the samples in the output tensor on top of it. However, the introduction of shortcut connections implies that $\mathfrak{Z}^l$ may correspond to two or even more tensors on top of it. As shown in Fig. 5, for direct shortcut connections, the pruning in $\mathfrak{Z}^2$ relies on $\mathfrak{Z}^3$ and $\mathfrak{Z}^4$, and $\mathfrak{Z}^4$ in turns replies on $\mathfrak{Z}^5$ and $\mathfrak{Z}^6$. for transformed shortcut connections, both $\mathfrak{Z}^{16}$ and $\mathfrak{Z}_t^{16}$ rely on $\mathfrak{Z}^{17}$ and $\mathfrak{Z}_t^{18}$. Therefore, ThiNet is not suitable for shortcut connections. As a result, a hybrid channel pruning criterion is employed in our proposed approach:

For those convolutional layers not involved in shortcut connections, ThiNet scheme is applied to them to determine the shrinking rates of their output tensors. As demonstrated in Fig. 5, $\mathfrak{Z}^3$, $\mathfrak{Z}^5$, $\mathfrak{Z}^{15}$ and $\mathfrak{Z}^{17}$ are not involved in shortcut connections. ThinNet scheme is applied to them and their shrinking rates are determined according to the output tensors on top of them, $\mathfrak{Z}^4$, $\mathfrak{Z}^6$, $\mathfrak{Z}^{16}$ and $\mathfrak{Z}^{18}$, respectively.

For those relevant convolutional layers linked via direct shortcut connections, they are handled altogether with a post-processing mechanism. l_1-norm based scheme [34] is applied to the output tensor of the lowest convolutional layer and determine its shrinking rate. The same shrinking rate is assigned to the output tensor of the rest relevant layers. Take SRNet for example, $\mathfrak{Z}^2$, $\mathfrak{Z}^4$, $\mathfrak{Z}^6$, and even $\mathfrak{Z}^8$, $\mathfrak{Z}^{10}$, $\mathfrak{Z}^{12}$ (do not appear in Fig. 5) are linked via direct shortcut connections. l_1-norm based scheme is applied to $\mathfrak{Z}^2$, the lowest one. Then the obtained shrinking rate is assigned to all the linked output tensors.

For those transformed shortcut connections, the two convolutional layers relevant to the element-wise addition in transformed shortcut connections are handled together. Given a main-branch output tensor $\mathfrak{Z}^l$, its shrinking rate is determined via l_1-norm based scheme. The same shrinking rate is assigned to $\mathfrak{Z}_t^l$, the output tensor in the corresponding transformed shortcut connection as well. Also take SRNet for example, l_1-norm based scheme is used to determine the shrinking rate of $\mathfrak{Z}^{16}$ and $\mathfrak{Z}^{18}$, and then the obtained shrinking rate is assigned to $\mathfrak{Z}_t^{16}$ and $\mathfrak{Z}_t^{18}$, respectively.

3) Channel-pruning-assisted architecture search: Please note that in CALPA-NET, channel pruning approaches is used to assist the determination of the shrinking rate of every involved convolutional layer. A new term "shrinking rate" is introduced in order to highlight the difference between our proposed CALPA-NET and existing channel pruning approaches. Given a convolutional layer L_l with a determined shrinking rate ζ_l, the corresponding pruning rate used in channel pruning approaches is $\gamma_l = 1 - \zeta_l$.

As mentioned in Sect. II-C1, the well-trained model N is traversed from bottom to top for every inclusive convolutional layer L_l with shrinking rate undetermined. The detailed algorithm is shown in Algorithm 1.

Algorithm 1 Shrinking rate determination algorithm for L_l and all the relevant convolutional layers.

```
Require: A well-trained model $N$, a standalone validation dataset $D_v$, a pre-defined step $\epsilon$ (set to 5% in our experiments), and a tolerable accuracy lost $\varsigma$ (set to 5% in our experiments).
Initialize the pruning rate $\gamma_l = 0\%$.
Validate $N$ in $D_v$. Denote the obtained validation accuracy as $Acc_0$. Let $Acc_{p_1} = Acc_0$.
loop
    Set $\gamma_l = \gamma_l + \epsilon$.
    if $L_l$ is not in shortcut connections then
        use ThiNet to prune $L_l$.
    else
        use $l_1$-norm based scheme to prune $L_l$ and all the relevant convolutional layers linked with it via shortcut connections.
    end if
    Let $N_p$ denotes the pruned model. Validate $N_p$ in $D_v$ and denote the validation accuracy as $Acc_{p_2}$.
    if $Acc_{p_1} - Acc_{p_2} > \varsigma$ then
        an obvious accuracy decline is observed.
        break
    else if $Acc_0 - Acc_{p_2} > \varsigma$ then
        the detection accuracy has gradually declined to beyond the tolerance threshold.
        break
    end if
    Let $Acc_{p_1} = Acc_{p_2}$.
end loop
set $\gamma_l = \gamma_l - \epsilon$, namely roll back to prior pruning rate.
Set the corresponding shrinking rate $\zeta_l = 100\% - \gamma_l$. If $L_l$ is in shortcut connections, the shrinking rates of all the relevant convolutional layers linked with it via shortcut connections are set to $\zeta_l$ as well.
```

After the bottom-up traversal, every inclusive convolutional layer L_l is assigned a determined shrinking rate ζ_l. The resulting CALPA-N model is obtained via shrinking the volume of $\mathfrak{Z}^l$, the before-activation output tensor of every inclusive L_l to $\mathbb{R}^{K^l_\zeta \times H^l \times W^l}$, in which $K^l_\zeta = K^l \cdot \zeta_l$. The obtained CALPA-$N$ is reset and trained from scratch.

III. Experiments

A. Experiment setup

The primary image dataset used in our experiments is the union of BOSSBase v1.01 [41] and BOWS2 [42][3], each of which contains 10,000 512 × 512 grayscale spatial images. All of the images were resized to 256 × 256 using Matlab® function *imresize*. The corresponding JPEG images were further generated with QFs (Quality Factors) 75 and 95. In every experiments, 10,000 BOWS2 images and 4,000 randomly selected BOSSBase images were used for training. Another 1,000 randomly selected BOSSBase images were for validation. The remaining 5,000 BOSSBase images were retained for testing. Furthermore, following the generation pipeline mentioned in [26], a subset of ImageNet CLS-LOC [43] dataset with 250,000 grayscale 256 × 256 JPEG images was also introduced to demonstrate the performance of CALPA-NET on large-scale dataset under cover source diversity scenario. ALASKA [44] is another large-scale dataset introduced to evaluate the performance of CALPA-NET under stego source diversity scenario. Its JPEG compressed (with QF 75) grayscale 256 × 256 version (v2) was adopted [4].

Four representative steganographic schemes, UERD [5] and J-UNIWARD [6] for JPEG domain, and HILL [3] and S-UNIWARD [6] for spatial domain, were our attacking targets in the experiments. For JPEG steganographic algorithms, the embedding payloads were set to 0.2 and 0.4 bpnzAC (bits per non-zero AC DCT coefficient). For spatial domain steganographic algorithms, the embedding payloads were set to 0.2 and 0.4 bpp (bits per pixel).

XuNet2 [24] and SRNet [26] were selected as the two initial architectures of our proposed CALPA-NET. Our implementation of CALPA-NET and its corresponding initial architectures are based on TensorFlow [45]. Unless otherwise specified, the two initial architectures were trained with the hyper-parameters mentioned in the corresponding original papers. The batch size in the training procedure was set to 32 (namely 16 cover-stego pairs). The maximum number of iterations was set to 50×10^4 for SRNet, and 32×10^4 for XuNet2. CALPA-XuNet2 and CALPA-SRNet adopted the same maximum number of iterations as their corresponding initial architectures. Like their corresponding initial architectures, the optimizers and learning rate schedules used for CALPA-SRNet and CALPA-XuNet2 were specifically defined as follows: CALPA-SRNet were trained using Adamax with learning rate starting from 0.001. After 40×10^4 iterations the learning rate was reduced to 0.0001. CALPA-XuNet2 was trained using mini-batch stochastic gradient descent with learning rate starting from 0.001. The learning rate was set to decrease 10% for every 5,000 iterations. The momentum was fixed to 0.9.

In each experiment, unless otherwise specified, the model was validated and saved every one epoch (in primary dataset, $14,000/16 = 875$ iterations). The one with the best validation accuracy was evaluated on the corresponding testing set. All of the experiments were conducted on a GPU cluster with sixteen NVIDIA® Tesla® P100 GPU cards. Bounded by computational resources, every experiment was repeated three times, and the mean of the results on testing set were reported. The source codes and auxiliary materials are available for download from GitHub [5].

B. Compactness of CALPA-NET

In this section, we take CALPA-SRNet as example to analyze its compactness and effectiveness.

1) Compactness of CALPA-NET with different ς: In Tab. I, we compare the effect of different tolerable accuracy lost ς

[3] For a fair comparison, we tried our best to adopt almost the same experiment setup as which SRNet [26] was evaluated on.

[4] http://alaska.utt.fr/ALASKA_v2_JPG_256_QF75_GrayScale.sh

[5] https://github.com/tansq/CALPA-NET

TABLE I

Effect of different tolerable accuracy lost ς on the parameters, FLOPs, and detection accuracies of CALPA-SRNet. The detection accuracies were obtained on the stand-alone testing set. The Two training scenarios are compared: "trained from scratch" and "finetuned". The trained models are aiming at detecting J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75. The percentages of parameters and FLOPs of CALPA-SRNet compared to original SRNet are listed in parentheses. For accuracies, the terms in parentheses with preceding ↓ and ↑ denote decrement or increment compared to original SRNet.

	Original SRNet	**CALPA-SRNet with a given tolerable accuracy lost ς**				
		2%	5%	10%	20%	50%
Parameters	477.06×10^4	8.43×10^4 **(1.77%)**	6.93×10^4 **(1.45%)**	4.71×10^4 **(0.99%)**	3.45×10^4 **(0.72%)**	2.24×10^4 **(0.47%)**
FLOPs	5.95×10^9	2.29×10^9 **(38.5%)**	1.97×10^9 **(33.1%)**	1.54×10^9 **(25.9%)**	1.37×10^9 **(23.1%)**	0.75×10^9 **(12.7%)**
				Trained from scratch		
Accuracy	92.98%	93.12% (**↑0.14%**)	92.86% (**↓0.12%**)	92.63% (**↓0.35%**)	92.36% (**↓0.62%**)	87.94% (**↓5.04%**)
				Finetuned		
		93.07% (**↑0.09%**)	92.8% (**↓0.18%**)	92.84% (**↓0.14%**)	92.59% (**↓0.39%**)	87.63% (**↓5.35%**)

on the model parameters, FLOPs, and detection accuracies of CALPA-SRNet.

During calculation, the number of parameters and the number of FLOPs for every convolutional layer are determined as follows (let | • | denotes the number of the corresponding variable): |parameters|=|input channels|×|kernel size|×|output channels|; |FLOPs|=|output channel size|×|parameters|.

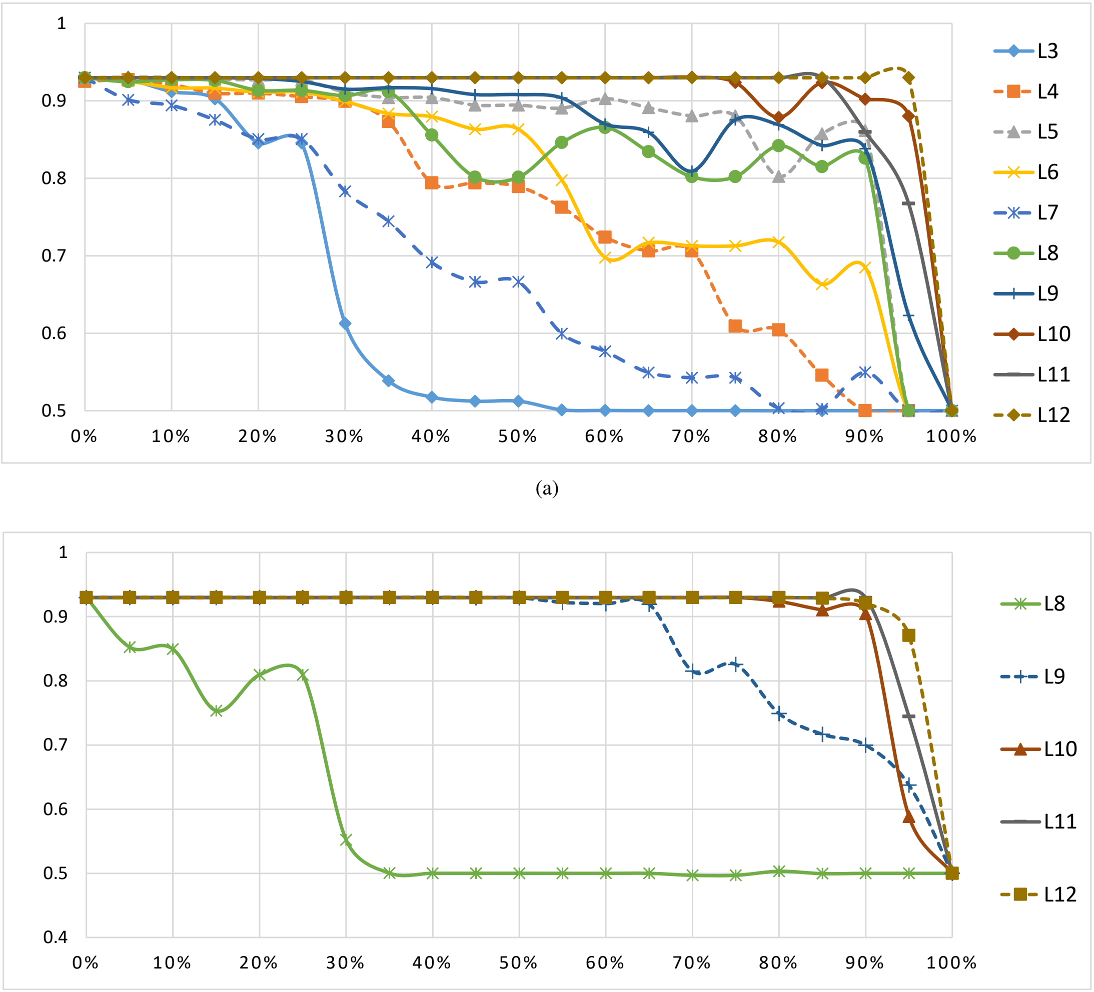
Here our adopted "trained from scratch" scenario is compared with the traditional "training-pruning-finetuning" pipeline. In "training-pruning-finetuning" pipeline, the weights in non-pruned kernels of the model were kept. Then the pruned model was re-trained/finetuned for another 50×10^4 iterations. From Tab. I we can see that with the rise of tolerable accuracy lost ς, the parameters and FLOPs required by CALPA-SRNet significantly decrease. However, with a mild ς (2% ~ 20%), the detection accuracy of the shrunken model is close to the original SRNet. Another notable thing is that the detection performances achieved by the shrunken models trained from scratch are almost equivalent to those with "training-pruning-finetuning" pipeline. Such a phenomenon indicates that in CALPA-Net, the efficient network architecture obtained via channel-pruning-assisted search is critical.

By contrasting the benefits of the reduction of parameters and FLOPs with the losses of detection performance, ς is set to 5% in our final proposed framework. Please note that when ς = 5%, CALPA-SRNet achieves similar detection performance as the original SRNet with mere 1.45% parameters (6.93×10^4 vs. 477.06×10^4) and 33.1% FLOPs (1.97×10^9 vs. 5.95×10^9). The corresponding architecture of CALPA-SRNet can be referred back to Fig. 1. From Fig. 1 we can see CALPA-SRNet presents a slender bottleneck-like structure in contrast to traditional deep-learning based steganalyzers.

However, please keep in mind that besides the complexity of the deep-learning model itself, there are quite a few other factors which influence the training efficiency, including access speed of the hard disk where the data is stored in, size of the hard disk cache/memory cache, speed of the high-bandwidth bridge on the motherboard. In order to maximize the benefits offered by CALPA-SRNet, high-end hardwares such as fast SSDs (solid-state drives) are highly recommended. Furthermore, please note that during the training procedure, the overwhelming majority of GPU memory allocation is used to store the input image batch and the corresponding input/output channels of all layers, rather than the model parameters. Therefore the superiority of CALPA-NET is its significantly reduced parameters and FLOPs, not smaller GPU memory allocation.

2) Tolerance to pruning rate in our approach: In Fig. 6, we show how the validation accuracy changes with successive growing pruning rates when determining the shrinking rate of every inclusive convolutional layer of CALPA-SRNet. In Algorithm 1, we just run the loop (from Line 3 to Line 19) for γ_l from 0% till 100%, and disable the early exit conditions from Line 11 to Line 17 to obtain the corresponding validation accuracies. From Fig. 6 no particular patterns can be observed. However, no matter for ThiNet scheme (as shown in Fig. 6(a)), or for l_1-norm based scheme (as shown in Fig. 6(b)), convolutional layers in top blocks always present high tolerance to severe pruning rates.

3) Effectiveness of CALPA-NET compared to traditional "training-pruning-finetuning" pipeline: In Fig. 7, we show how the training accuracy and validation accuracy change with successive training iterations for the original SRNet, the corresponding CALPA-SRNet trained from scratch, and the pruned SRNet with "training-pruning-finetuning" pipeline. When trained to detect J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75, SRNet achieved the best validation accuracy at 42.35×10^4 iterations. The corresponding model was selected out to apply channel-pruning-assisted search to get CALPA-SRNet. CALPA-SRNet was reset and trained from scratch. As shown in Fig. 7, the validation accuracies of CALPA-SRNet trained from scratch had become stable at around 40×10^4 iterations. For comparison, training procedure of the pruned SRNet was resumed. It was further finetuned till 80×10^4 iterations. But it can be observed that the validation accuracies of pruned SRNet during the finetuning procedure was unstable. Furthermore, its best validation accuracy did not surpass CALPA-SRNet trained from scratch.



(a)

(b)

Fig. 6. Validation accuracies vs. growing pruning rates when determining the shrinking rate of every inclusive convolutional layer of CALPA-SRNet. The trained models are aiming at detecting J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75. $\varsigma = 5$. (a) is for ThiNet scheme in blocks "L3"—"L12". (b) is for l_1-norm based scheme in transformed shortcut connections of blocks "L8"—"L12".

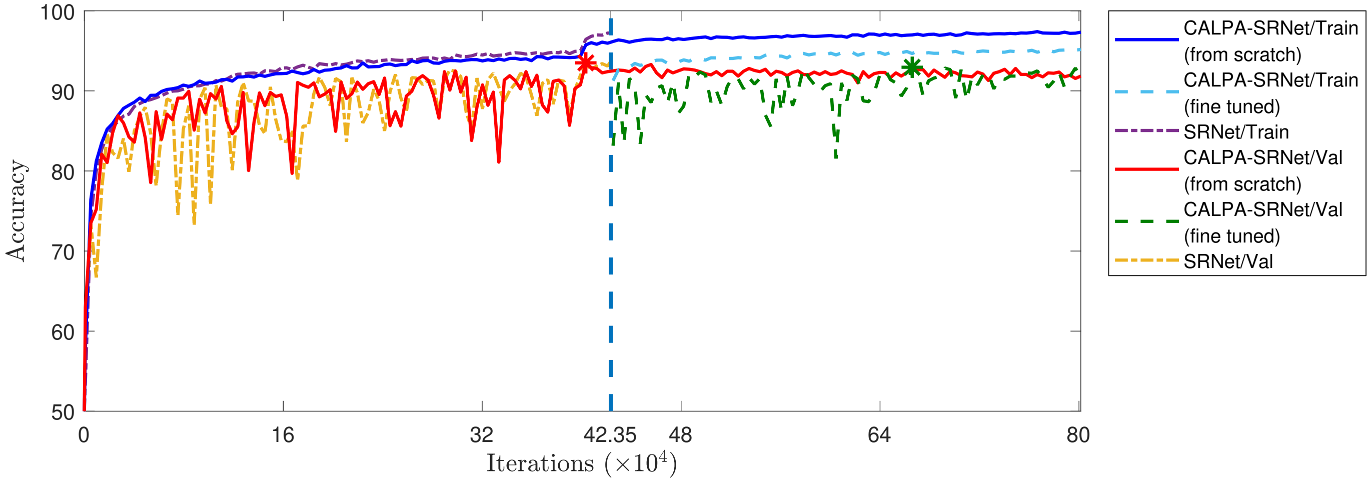
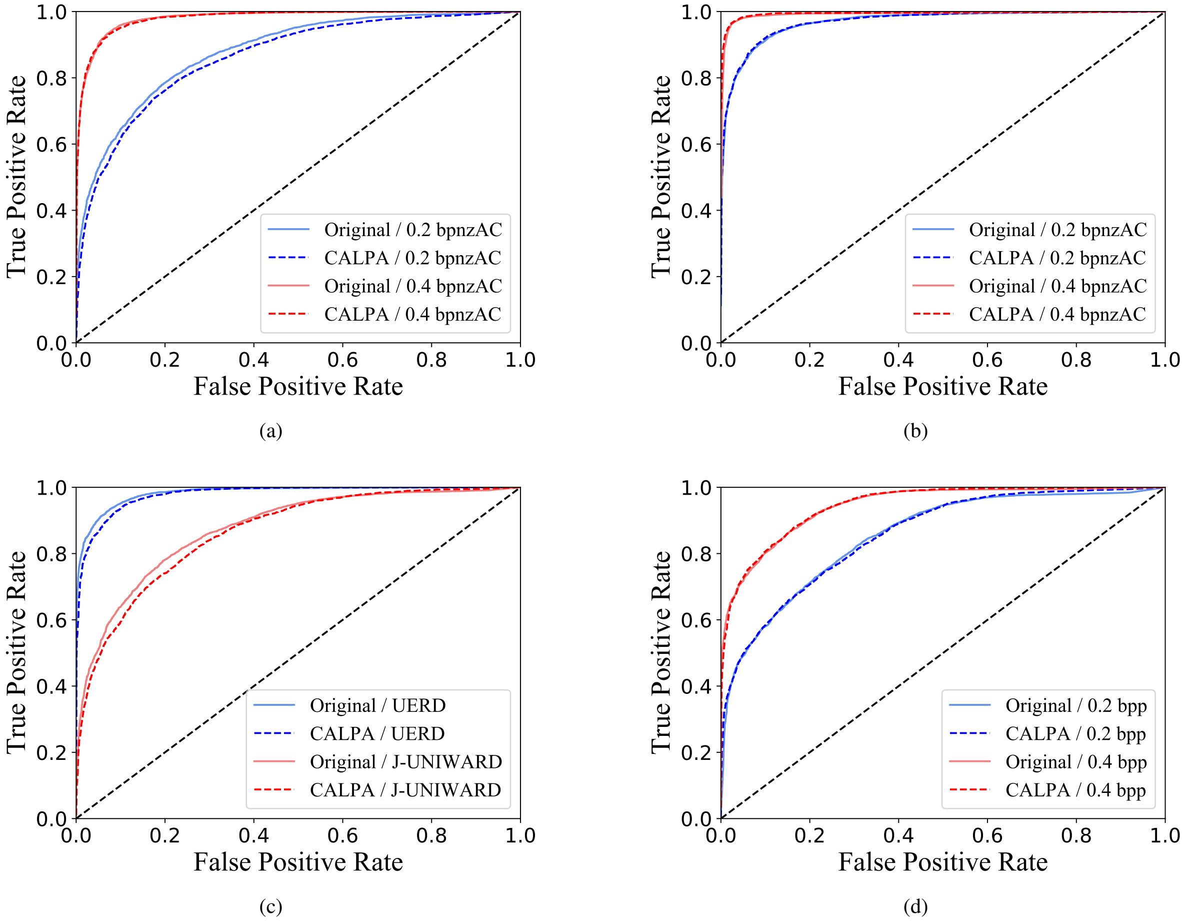


Fig. 7. Comparison of the training accuracies, validation accuracies and testing accuracies vs. training iterations for the original SRNet, CALPA-SRNet trained from scratch, and CALPA-SRNet with "training-pruning-finetuning" pipeline. The trained models are aiming at detecting J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75. $\varsigma = 5$. "✱" denotes the point with the highest value at the corresponding plot.



(a)

(b)

(c)

(d)

Fig. 8. Comparison of testing accuracy of CALPA-SRNet and the corresponding original SRNet. (a) The trained models are aiming at detecting J-UNIWARD stego images with QF 75. (b) Aiming at detecting UERD stego images with QF 75. (c) Aiming at detecting UERD/J-UNIWARD stego images with 0.4 bpnzAC payload and QF 95. (d) Aiming at detecting HILL spatial-domain stego images with 0.2/0.4 bpp payload.

C. Detection performance of CALPA-NET

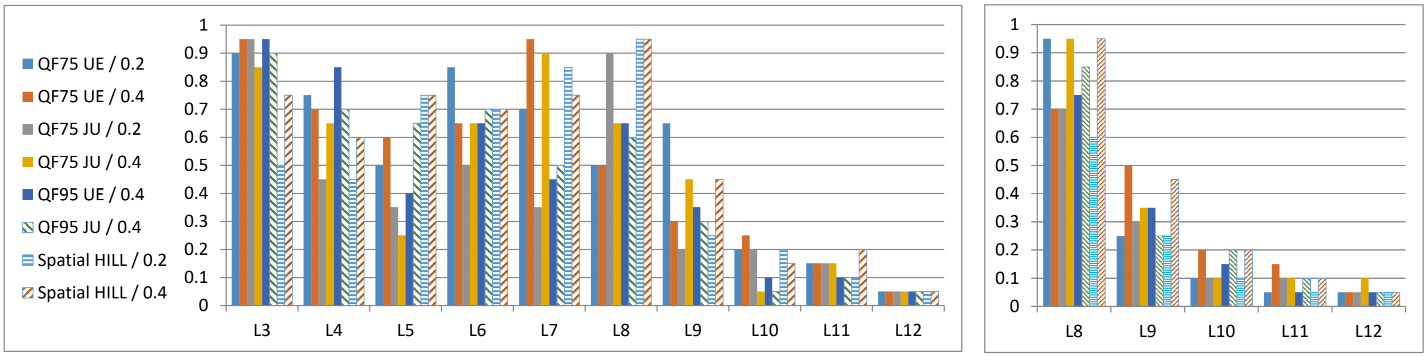
In Fig. 8, we compare the detection performance of CALPA-SRNet and the corresponding original SRNet. From Fig. 8(a) and Fig. 8(b) we can see that CALPA-SRNet obtains comparable detection performance when aiming at detecting JPEG stego images with QF 75 (drops a bit with J-UNIWARD/0.2 bpnzAC). The detection performance of CALPA-SRNet is slightly worse than original SRNet when JPEG stego images are with QF 95, as shown in Fig. 8(c). Fig. 8(d) indicates that when used to detect 0.4 bpp HILL spatial-domain stego images, CALPA-SRNet can also obtain almost the same detection performance. Please note that all the above high performance of CALPA-SRNet are achieved with mere 1%∼3% parameters compared to original SRNet.

In Fig. 9, we show the corresponding shrinking rates of CALPA-SRNets used in Fig. 8. From Fig. 9 we can observe that the shrinking rate of every convolutional layer adapts to the target steganographic scheme, the embedding payload, the embedding domain and even the actual quality factor used in JPEG target images. In general, the shrinking rate reduces as the level of the corresponding convolutional layer gets higher and higher. Significantly low shrinking rates can always be observed in top convolutional layers.

In Tab. II, we compare the detection performance of CALPA-XuNet2 and original XuNet2. We adopt an alternative performance measure used in [26]: the false-alarm rate for a given stego-image detection probability. For instance, $P_{FA}(30\%)$ denotes the false-alarm rate for 30% stego-image detection probability. From Tab. II, by looking into all the false-alarm rates with stego-image detection probability stepping from 30% to 50% and then to 70%, it is clear that on the whole, there is no distinction between the detection performance of CALPA-XuNet2 and that of XuNet2. The results demonstrate the amazing detection performance of CALPA-XuNet2 since it is quite a lightweight model compared to original XuNet2. The architecture of one of the CALPA-XuNet2 models aiming at detecting J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75 can be found in Fig. 1, in which we can see CALPA-XuNet2 also presents a slender bottleneck-like structure.

D. Adaptivity, Transferability, and Scalability of CALPA-NET

As demonstrated in prior experiments, the structure of CALPA-NET is adaptive to the actual targets it aims at due to the fact that CALPA-NET is data-driven. Firstly in Fig. 10, we evaluate the effectiveness of the adaptive data-driven scheme used in CALPA-NET taking CALPA-SRNet as example. Here two less adaptive alternative schemes are introduced:



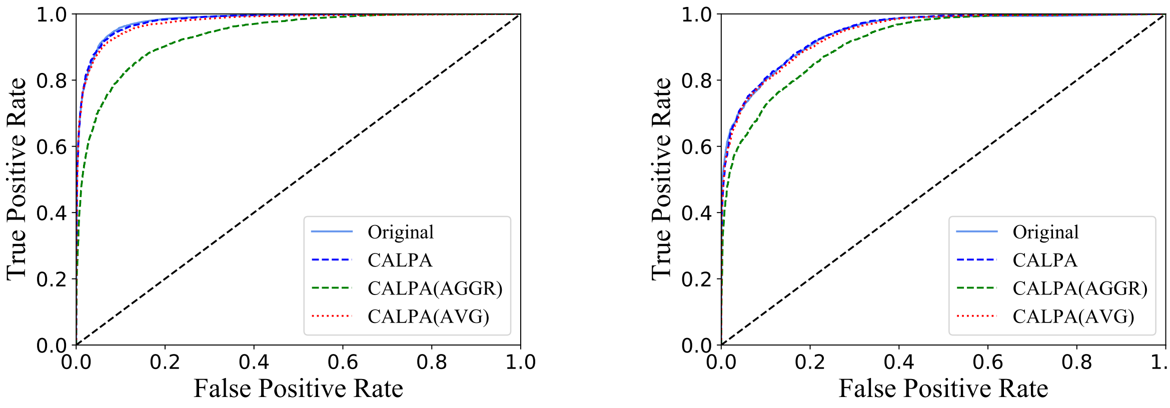
(a)

(b)

Fig. 9. The corresponding shrinking rates of CALPA-SRNets used in Fig. 8. (a) is for ThiNet scheme in blocks “L3”—“L12”. (b) is for l_1-norm based scheme in transformed shortcut connections of blocks “L8”—“L12”.

TABLE II
Comparison of detection performance of CALPA-XuNet2 and the corresponding original XuNet2. The detection performance is measured via false-alarm rate for a given stego-image detection probability. Results for J-UNIWARD/UERD stego images with 0.2/0.4 bpnzAC payload and QF 75/95 are included.

			CALPA-XuNet2			**XuNet2**		
Quality Factors	**Targets**		P_{FA}(70%)	P_{FA}(50%)	P_{FA}(30%)	P_{FA}(70%)	P_{FA}(50%)	P_{FA}(30%)
QF75	**J-UNIWARD**	0.2 bpnzAC	1.54%	0.30%	0.12%	1.06%	0.13%	0%
		0.4 bpnzAC	0.1%	0.02%	0.01%	0.15%	0.04%	0.02%
	UERD	0.2 bpnzAC	0.91%	0.12%	0.02%	0.7%	0.14%	0.04%
		0.4 bpnzAC	0%	0%	0%	0.07%	0%	0%
QF95	**J-UNIWARD**	0.2 bpnzAC	36.16%	17.64%	3.28%	41.36%	22.69%	7.06%
		0.4 bpnzAC	1.06%	0.06%	0%	0.72%	0.06%	0%
	UERD	0.2 bpnzAC	6.62%	0.66%	0.02%	4.64%	0.54%	0.06%
		0.4 bpnzAC	0.46%	0.09%	0.02%	0.46%	0.10%	0.05%



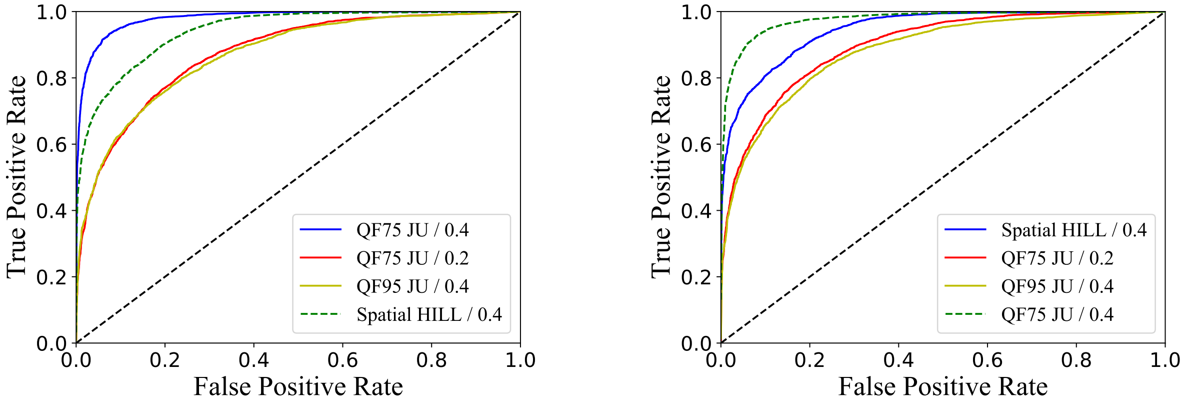
(a)

(b)

Fig. 10. Comparison of testing accuracy of CALPA-SRNet with two less adaptive alternative schemes, the AGGR scheme and the AVG scheme. (a) For J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75. (b)For HILL stego images with 0.4 bpp payload.

- AGGR scheme: The global shrinking rate of all the inclusive convolutional layers is aggressively set to the minimal one determined in the CALPA-NET bottom-up traversal.
- AVG scheme: The global shrinking rate of all the inclusive convolutional layers is set to the average of all the determined shrinking rates.

Refer back to the CALPA-SRNet illustrated in Fig. 1, the global shrinking rate can be obtained as 5% and 56% for AGGR scheme and AVG scheme respectively. From Fig. 10 we can see though AGGR scheme has the least parameters and FLOPs, its detection performance degrades remarkably. AVG scheme possesses similar scale of parameters and FLOPs as CALPA-SRNet, but no longer has bottleneck-like structure. Its detection performance is close to CALPA-SRNet. However, it is still a bit inferior to CALPA-SRNet when aiming at J-



(a)

(b)

Fig. 11. Detection performance of CALPA-SRNet when applied to another targets. “JU” in the legend is short for J-UNIWARD steganographic algorithm. (a) The architecture is original aiming at J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75. (b) The architecture is original aiming at HILL stego images with 0.4 bpp payload.

UNIWARD stego images.

Next, we investigate the transferability of the obtained architecture of CALPA-NET. Firstly we take the CALPA-SRNet architecture aiming at J-UNIWARD stego images with 0.4 bpnzAC payload and QF 75 as target. From Fig. 11(a) it is clear that the CALPA-SRNet architecture obtains good detection performance in the three mismatched scenarios. Furthermore, the specific JPEG-oriented CALPA-SRNet architecture remains effective for spatial-domain HILL stego images, implying that HILL and J-UNIWARD share some intrinsic characteristics although they are in completely different domain. Oppositely, we also take the CALPA-SRNet architecture aiming at HILL stego images in spatial domain with 0.4 bpp payload as target. Fig. 11(b) shows its detection performance on the three JPEG domain scenarios. We can see that the CALPA-SRNet architecture originally for spatial domain scenario is still effective on the three diverse JPEG domain scenarios, which implies that our proposed CALPA-NET approach presents a cross-domain universal characteristics.

We then investigate the transferability of the trained CALPA-NET model. In Tab. III we compare the detection performance of CALPA-SRNet and original SRNet trained on one target and tested on another target. In order to make a fair comparison, we used the following performance measurement (as in Tab. II of [26]): $P_E = \min_{P_{FA}} \frac{1}{2}(P_{FA} + P_{MD})$, where P_{FA} and P_{MD} are the false alarm rate and miss detection rate. From Tab. III it can be clearly observed that when the payload of the targets are the same, CALPA-SRNet achieves similar, and even better transferability compared with original SRNet.

At last the scalability of CALPA-NET is investigated. We firstly demonstrate the performance of CALPA-SRNet on the subset of ImageNet CLS-LOC dataset, which is of great cover source diversity. From Fig. 12 we can see even in a tenfold larger dataset, CALPA-SRNet still shows similar validation and testing performance compared to original SRNet. Please note that the gap between training accuracies and validation accuracies for original SRNet is much bigger than that for CALPA-SRNet, indicating that CALPA-SRNet may be less vulnerable to overfitting the training set.

Then we evaluate the performance of CALPA-SRNet on ALASKA, which is of great stego source diversity. For the 80,005 ALASKA images, the train/validation/test dataset partition is 35,000/5,000/40,005. Following the configuration in [44], three more JPEG steganographic algorithms, UED [46], EBS [47] and nsF5 [48] were used, alongside with J-UNIWARD. The probability of using each of the steganographic algorithms above was 30%, 15%, 15% and 40%, respectively. The corresponding embedding rate was 0.25 bpnzAC, 0.25 bpnzAC, 0.05 bpnzAC and 0.4 bpnzAC, respectively [6].

From Fig. 13(a), we can see SRNet achieved the best validation accuracy at 55.3311×10^4 iterations. The CALPA-SRNet architecture obtained from it could become stable at around 40×10^4 iteration when trained from scratch. The highest validation accuracy of CALPA-SRNet was at the same level as that of the original SRNet. Compared to Fig. 7, we can see that on ALASKA dataset, the validation accuracies of CALPA-SRNet trained from scratch are obviously higher than the pruned SRNet with traditional “training-pruning-finetuning” pipeline. For the sake of completeness, the testing accuracies of the three models with the best validation accuracy are also reported here—CALPA-SRNet: 78.73%; the pruned SRNet with “training-pruning-finetuning” pipeline: 77.53%; original SRNet: 78.42%.

The conceptual structure of the corresponding CALPA-SRNet model is illustrated in Fig. 13(b). Compared with the conceptual structure of CALPA-SRNet in Fig. 1, we can see that though there are certain differences, the CALPA-SRNet for ALASKA also presents a slender bottleneck-like structure. As a result, compared with original SRNet, it is only with 1.97% parameters (9.22×10^4) and 36.5% FLOPs (2.17×10^9).

E. Further discussion: why train CALPA-NETs from scratch

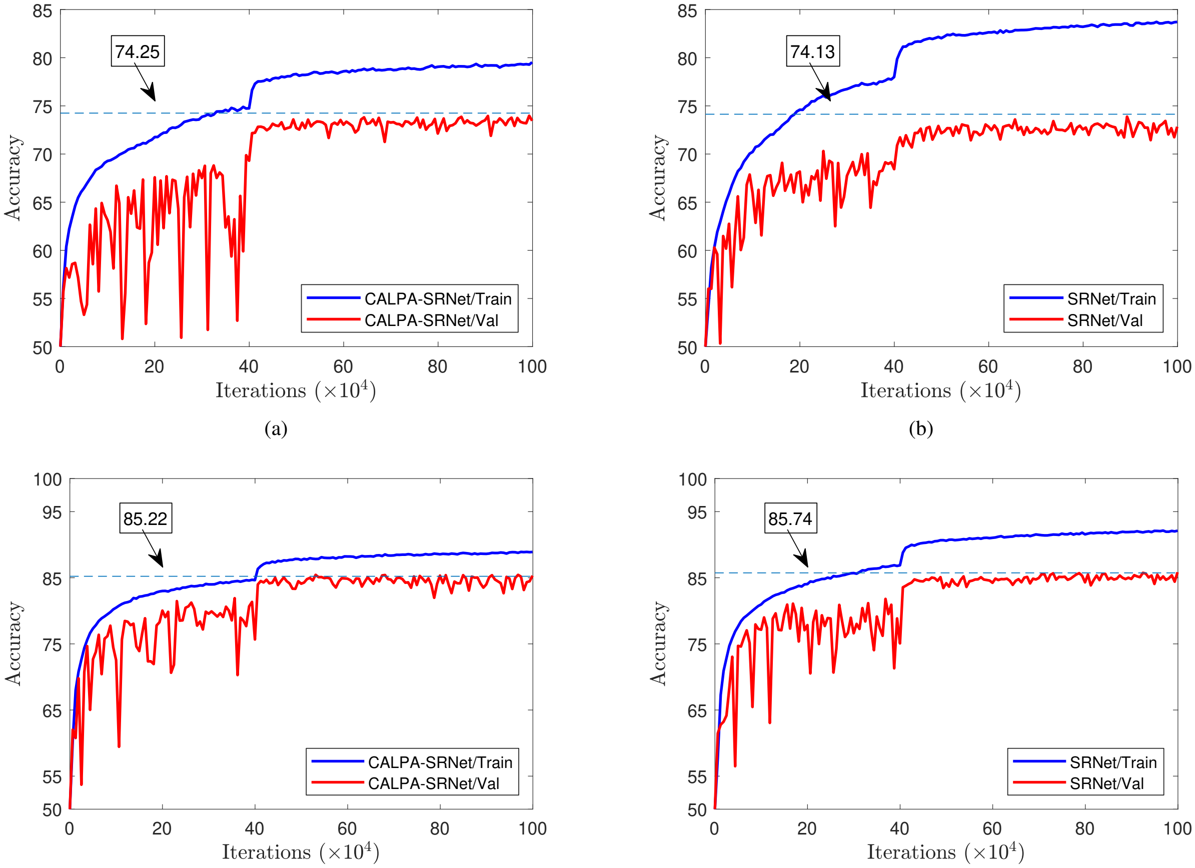
Putting all the experimental evidences together, the reasons for training CALPA-NETs from scratch are provided as fol-

[6] The default embedding rates in the ALASKA embedding script were adopted since the logs of processing pipelines have not been provided for ALASKA v2.

TABLE III

DETECTION PERFORMANCE OF CALPA-SRNET AND ORIGINAL SRNET TRAINED ON ONE TARGET AND THEN TESTED ON ANOTHER TARGET. THE PAYLOAD OF THE TARGETS WERE FIXED TO 0.4 BPNZAC/BPP. THE DETECTION PERFORMANCE WAS MEASURED WITH TOTAL ERROR PROBABILITY P_E.

JPEG domain			**Spatial domain**		
CALPA-SRNet					
Tested on Trained on	J-UNIWARD	UERD	Tested on Trained on	S-UNIWARD	HILL
J-UNIWARD	7.5%	5.64%	S-UNIWARD	10.63%	26.03%
UERD	24.2%	3.17%	HILL	23.89%	14.63%
SRNet					
Tested on Trained on	J-UNIWARD	UERD	Tested on Trained on	S-UNIWARD	HILL
J-UNIWARD	7.02%	5.35%	S-UNIWARD	12.75%	26.99%
UERD	26.04%	3.37%	HILL	22.48%	14.7%



(a)

(b)

(c)

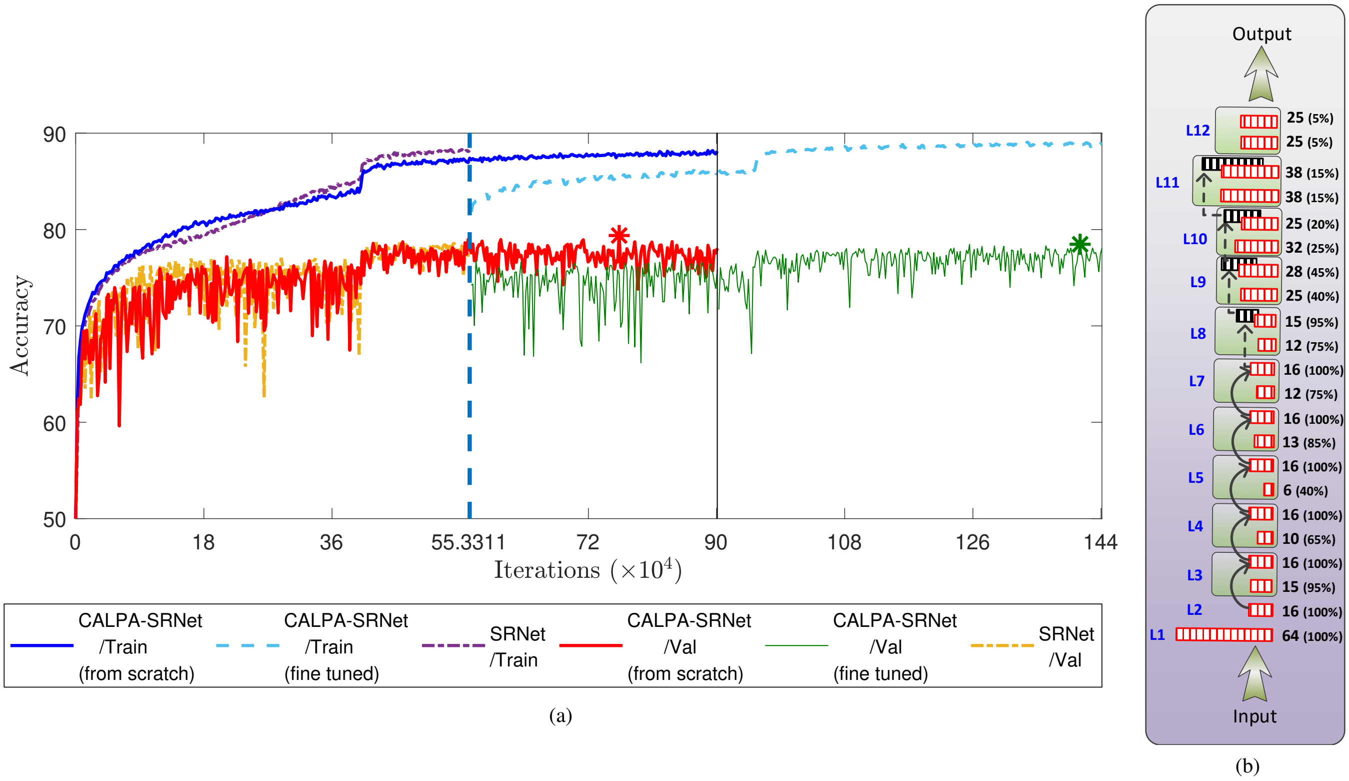
(d)

Fig. 12. Comparison of the training accuracies, validation accuracies and testing accuracies vs. training iterations for CALPA-SRNet and the corresponding original SRNet on a subset of CLS-LOC dataset. The trained models are aiming at detecting J-UNIWARD/UERD stego images with 0.4 bpnzAC payload and QF 75. The blue dashed line in every sub-figure marks the highest testing accuracy achieved by the trained model. (a) CALPA-SRNet, aiming at J-UNIWARD; (b) Original SRNet, aiming at J-UNIWARD; (c) CALPA-SRNet, aiming at UERD; (d) Original SRNet, aiming at UERD.

lows:

Firstly, in view of detection performance, training CALPA-NETs from scratch is better than finetuning it. In Tab. I, it is clear that the detection performance of CALPA-SRNet trained from scratch is slightly better than finetuning it on BOSSBase+BOWS2 dataset, when tolerable accuracy lost ς is low. On ALASKA dataset, a larger and more diverse dataset, the detection performance of CALPA-SRNet trained from scratch is obviously better than the pruned SRNet with "training-pruning-finetuning" pipeline.

Secondly, in view of training efficiency, training CALPA-NETs from scratch is better than finetuning it. Fig. 7 shows that on BOSSBase+BOWS2 dataset, the validation accuracies of pruned SRNet during the finetuning procedure was unstable.



(a)

(b)

Fig. 13. (a) Comparison of the training accuracies and validation accuracies vs. training iterations for the original SRNet, CALPA-SRNet trained from scratch, and CALPA-SRNet with "training-pruning-finetuning" pipeline on ALASKA dataset. "✱" denotes the point with the highest value at the corresponding plot. (b) The conceptual structure of the corresponding CALPA-SRNet model.

We can see more distinct efficiency difference on the ALASKA dataset. As shown in Fig. 13(a), even the best validation obtained by the pruned SRNet at further 85.7304×10^4 iterations $((85.7304{+}55.3311){\times}10^4 = 141.0615{\times}10^4)$ was inferior. As comparison, the validation accuracies of CALPA-SRNet trained from scratch became stable at around $40{\times}10^4$ iterations and achieved the peak at mere 76.3263×10^4 iterations.

Thirdly and the most importantly, the effectiveness of CALPA-NETs trained from scratch demonstrates what matters most is the cost-effective architecture obtained by CALPA-NET approach, rather than the preserved parameters in the pruned steganalytic models. Furthermore, Fig. 11 demonstrates that CALPA-NET architecture presents general applicability. Fig. 13(a) also demonstrates that there exists an effective slender bottleneck-like CALPA-SRNet architecture under the stego source diversity scenario. Therefore, the users can absolutely apply one versatile CALPA-NET architecture to multiple steganalytic scenarios and train it from scratch adaptively with limited budget. Such an ability is impossible for traditional three-stage training-pruning-finetuning pipeline.

IV. Concluding remarks

In this paper we propose CALPA-NET, which is aiming at adaptively search efficient network structure on top of existing over-parameterized and vast deep-learning based steganalyzers. The major contributions of this work are as follows:

- We have observed that the broad inverted-pyramid structure of existing deep-learning based steganalyzers cannot boost detection performance even with fifty fold parameter. A theoretical reflection is proposed to argue that such structure might contradict the well-established model diversity oriented philosophy.
- We have proposed a channel-pruning-assisted deep residual network architecture search approach which uses a hybrid criterion combined with ThiNet and l_1-norm based network pruning scheme. In the proposed network architecture, the traditional training-pruning-finetuning network pruning pipeline is completed abandoned.
- The extensive experiments conducted on de-facto benchmarking image datasets show that CALAP-NET can achieve comparative detection performance with just a few percent of the model size and a small proportion of computational cost.

Our future work will focus on two aspects: (1) development of a fast adaptive structural adjustment algorithm to make deep-learning based steganalyzers self adapt to targets without the introduction of redundant parameters/components; (2) further exploration of the feasibility of completely automatic deep-learning based steganalytic framework generation.

Appendix A
Theoretical reflection

In the CNN pipeline, given a convolutional layer L_l, let us start from (1). Let $\mathcal{O}_{jk::} = \widetilde{\mathcal{Z}}^{l-1}_{j::} * \mathcal{W}^{l}_{jk::}$, then $\mathcal{Z}^{l}_{k::} = \sum_{j=1}^{J} \mathcal{O}_{jk::}$, $1 \leq k \leq K^{l}$. Assume that $\mathcal{O}_{jk::}$ is a discrete sample drawn from a continuous latent distribution $\overline{O}_{jk}$, with PDF (Probability Distribution Function) o_{jk}. Accordingly, let $\mathfrak{Z}^l_{k::}$ be a sample of the latent distribution $\overline{Z}_k$ with PDF z_k. Please note that the PDF of the aggregation is equal to the convolution of the PDF of the aggregated terms:

$$z_k = \underbrace{o_{1k} * o_{2k} * \cdots * o_{jk} * \cdots * o_{Jk}}_{1 \leq j \leq J} \tag{3}$$

Denote the Fourier transform of o_{jk} and z_k as O_{jk} and $\mathcal{Z}_k$, respectively. Therefore in frequency domain, we can get:

$$\mathcal{Z}_k(\omega) = \prod_{j=1}^{J} O_{jk}(\omega) \tag{4}$$

Given two frequencies ω_1 and ω_2, assume that for a fixed k, O_{jk} in ω_1 contains sparse intensity fluctuations, while in ω_2 contains strong intensities. Therefore for most j, $|O_{jk}(\omega_1)| \ll |O_{jk}(\omega_2)|$. Obviously, for large enough J:

$$\frac{|\mathcal{Z}_k(\omega_1)|}{|\mathcal{Z}_k(\omega_2)|} = \frac{\left|\prod_{j=1}^{J} O_{jk}(\omega_1)\right|}{\left|\prod_{j=1}^{J} O_{jk}(\omega_2)\right|} = \frac{\prod_{j=1}^{J} |O_{jk}(\omega_1)|}{\prod_{j=1}^{J} |O_{jk}(\omega_2)|} \longrightarrow 0 \tag{5}$$

From (5) we can see the aggregation in (1) with large enough J suppresses sparse intensity fluctuations as well as heightens strong intensities in low-frequency subbands of $\mathfrak{Z}^l_{k::}$, $1 \leq k \leq K^l$.

References

[1] R. Böhme, *Advanced Statistical Steganalysis*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[2] T. Filler and J. Fridrich, "Gibbs construction in steganography," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 705–720, 2010.

[3] B. Li, M. Wang, J. Huang, and X. Li, "A new cost function for spatial image steganography," in *Proc. IEEE 2014 International Conference on Image Processing, (ICIP'2014)*, 2014, pp. 4206–4210.

[4] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 221–234, 2016.

[5] L. Guo, J. Ni, W. Su, C. Tang, and Y. Q. Shi, "Using statistical image model for JPEG steganography: Uniform embedding revisited," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2669–2680, 2015.

[6] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP Journal on Information Security*, vol. 2014, no. 1, pp. 1–13, 2014.

[7] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 2012 in neural information processing systems, (NIPS'2012)*, 2012, pp. 1097–1105.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[10] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 2015)*, 2015, pp. 1–9.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 2016)*, 2016, pp. 770–778.

[12] J. Fridrich and J. Kodovský, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012.

[13] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, "Selection-channel-aware rich model for steganalysis of digital images," in *Proc. 6th IEEE International Workshop on Information Forensic and Security (WIFS'2014)*, 2014, pp. 48–53.

[14] W. Tang, H. Li, W. Luo, and J. Huang, "Adaptive steganalysis based on embedding probabilities of pixels," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 734–745, 2016.

[15] S. Tan, H. Zhang, B. Li, and J. Huang, "Pixel-decimation-assisted steganalysis of synchronize-embedding-changes steganography," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 7, pp. 1658–1670, 2017.

[16] B. Li, Z. Li, S. Zhou, S. Tan, and X. Zhang, "New steganalytic features for spatial image steganography based on derivative filters and threshold LBP operator," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1242–1257, 2018.

[17] J. Kodovský and J. Fridrich, "Ensemble classifiers for steganalysis of digital media," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, 2012.

[18] S. Tan and B. Li, "Stacked convolutional auto-encoders for steganalysis of digital images," in *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA'2014)*, 2014, pp. 1–4.

[19] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in *Proc. IS&T/SPIE Electronic Imaging 2015 (Media Watermarking, Security, and Forensics)*, 2015, pp. 94 090J–1–94 090J–10.

[20] L. Pibre, P. Jérôme, D. Ienco, and M. Chaumont, "Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch," in *Proc. Media Watermarking, Security, and Forensics, Part of IS&T International Symposium on Electronic Imaging (EI'2016)*, 2016, pp. 1–11.

[21] G. Xu, H. Z. Wu, and Y. Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, 2016.

[22] J. Ye, J. Ni, and Y. Yi, "Deep learning hierarchical representations for image steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2545–2557, 2017.

[23] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich, "JPEG-phase-aware convolutional neural network for steganalysis of JPEG images," in *Proc. 5th ACM Information Hiding and Multimedia Security Workshop (IH&MMSec'2017)*, 2017, pp. 75–84.

[24] G. Xu, "Deep convolutional neural network to detect J-UNIWARD," in *Proc. 5th ACM Information Hiding and Multimedia Security Workshop (IH&MMSec'2017)*, 2017, pp. 67–73.

[25] J. Zeng, S. Tan, B. Li, and J. Huang, "Large-scale JPEG steganalysis using hybrid deep-learning framework," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1242–1257, 2018.

[26] M. Boroumand, M. Chen, and J. Fridrich, "Deep residual network for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1181–1193, 2018.

[27] J. Zeng, S. Tan, G. Liu, B. Li, and J. Huang, "WISERNet: Wider separate-then-reunion network for steganalysis of color images," *IEEE Transactions on Information Forensics and Security*, 2019, doi: 10.1109/TIFS.2019.2904413.

[28] M. Chaumont, "Deep learning in steganography and steganalysis from 2015 to 2018," *arXiv preprint arXiv:1904.01444*, 2019.

[29] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[30] C. Liu *et al.*, "Progressive neural architecture search," in *Proc. 2018 European Conference on Computer Vision (ECCV'2018)*, 2018, pp. 19–34.

[31] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Proc. 1990 Advances in neural information processing systems, (NIPS'1990)*, 1990, pp. 598–605.

[32] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. 2015 Advances in neural information processing systems, (NIPS'2015)*, 2015, pp. 1135–1143.

[33] H. Hu, R. Peng, Y.-W. Tai, and C.-K. Tang, "Network trimming: A data-driven neuron pruning approach towards efficient deep architectures," *arXiv preprint arXiv:1607.03250*, 2016.

[34] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets," in *Proc. 2017 International Conference on Learning Representations, (ICLR'2017)*, 2017.

[35] J.-H. Luo, J. Wu, and W. Lin, "ThiNet: A filter level pruning method for deep neural network compression," in *Proc. 2017 IEEE international conference on computer vision, (ICCV'2017)*, 2017, pp. 5058–5066.

[36] Z. Huang and N. Wang, "Data-driven sparse structure selection for deep neural networks," in *Proc. 2018 European Conference on Computer Vision (ECCV'2018)*, 2018, pp. 304–320.

[37] A. Veit, M. Wilber, and S. Belongie, "Residual networks behave like ensembles of relatively shallow networks," in *Proc. 2016 Advances in neural information processing systems, (NIPS'2016)*, 2016, pp. 550–558.

[38] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 2014)*, 2014, pp. 806–813.

[39] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 2014)*, 2014, pp. 1717–1724.

[40] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," in *Proc. 2019 International Conference on Learning Representations, (ICLR'2019)*, 2019.

[41] P. Bas, T. Filler, and T. Pevný, "Break our steganographic system—the ins and outs of organizing BOSS," in *Proc. 13th Information Hiding Workshop (IH'2011)*, 2011, pp. 59–70.

[42] P. Bas and T. Furon, "BOWS-2," http://bows2.ec-lille.fr, accessed: 2019-08-06.

[43] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[44] R. Cogranne, Q. Giboulot, and P. Bas, "The ALASKA steganalysis challenge: A first step towards steganalysis," in *Proc. 7th ACM Information Hiding and Multimedia Security Workshop (IH&MMSec'2019)*, 2019, pp. 125–137.

[45] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[46] L. Guo, J. Ni, and Y. Q. Shi, "An efficient JPEG steganographic scheme using uniform embedding," in *Proc. 4th IEEE International Workshop on Information Forensic and Security (WIFS'2012)*, 2012, pp. 169–174.

[47] C. Wang and J. Ni, "An efficient JPEG steganographic scheme based on the block entropy of DCT coefficients," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'2012)*. IEEE, 2012, pp. 1785–1788.

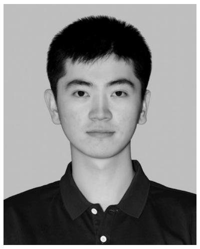
[48] J. Fridrich, T. Pevnỳ, and J. Kodovskỳ, "Statistically undetectable JPEG steganography: dead ends challenges, and opportunities," in *Proc. 9th Workshop on Multimedia & Security*, 2007, pp. 3–14.

Shunquan Tan (M'10–SM'17) received the B.S. degree in computational mathematics and applied software and the Ph.D. degree in computer software and theory from Sun Yat-sen University, Guangzhou, China, in 2002 and 2007, respectively.

He was a Visiting Scholar with New Jersey Institute of Technology, Newark, NJ, USA, from 2005 to 2006. He is currently an Associate Professor with College of Computer Science and Software Engineering, Shenzhen University, China, which he joined in 2007. His current research interests include multimedia security, multimedia forensics, and machine learning.

Weilong Wu received the B.S degree of computer science and technology from Guangdong Ocean University, Zhanjiang, China in 2018. He is currently a master student in Shenzhen University majoring in software engineering. His current research interests include multimedia forensics, model compression, neural architecture search and deep learning.

Zilong Shao received the B.S degree of computer science and technology from Shenzhen University, Shenzhen, China in 2019. He is currently a master student in Shenzhen University majoring in computer technology. His current research interests include multimedia forensics, model compression, neural architecture search and deep learning.



Qiushi Li received the B.S degree of information and computing science from Harbin University of Science and Technology, Shenzhen, China in 2019. He is currently a master student in Shenzhen University majoring in computer science and technology. His current research interests include multimedia forensics, model compression and deep learning.

Bin Li (S'07-M'09-SM'17) received the B.E. degree in communication engineering and the Ph.D. degree in communication and information system from Sun Yat-sen University, Guangzhou, China, in 2004 and 2009, respectively.

He was a Visiting Scholar with New Jersey Institute of Technology, Newark, NJ, USA, from 2007 to 2008. He is currently a Professor with Shenzhen University, Shenzhen, China, where he joined in 2009. He is also the director of Shenzhen Key Laboratory of Media Security, the vice director of Guangdong Key Lab of Intelligent Information Processing, and a member of Peng Cheng Laboratory. He has served as the IEEE Information Forensics and Security TC member since 2019. His current research interests include multimedia forensics, image processing, and deep machine learning.

Jiwu Huang (M'98–SM'00–F'16) received the B.S. degree from Xidian University, Xian, China, in 1982, the M.S. degree from Tsinghua University, Beijing, China, in 1987, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 1998. He is currently a Professor with the College of Information Engineering, Shenzhen University, Shenzhen, China. His current research interests include multimedia forensics and security. He is a member IEEE Signal Processing Society Information Forensics and Security Technical Committee and serves as an Associate Editor for the IEEE Transactions on Information Forensics and Security. He was a General Co-Chair of the IEEE Workshop on Information Forensics and Security in 2013 and a TPC Co-Chair of the IEEE Workshop on Information Forensics and Security in 2018.